UNIVERSIDAD AUTÓNOMA METROPOLITANA
Unidad Iztapalapa

Practica #1

Sistemas Operativos

Olvera Monroy Gonzalo

<2173011224>

Prof. Orlando Muñoz Texzocotetla

Trimestre: 20 - O

Actividad #1

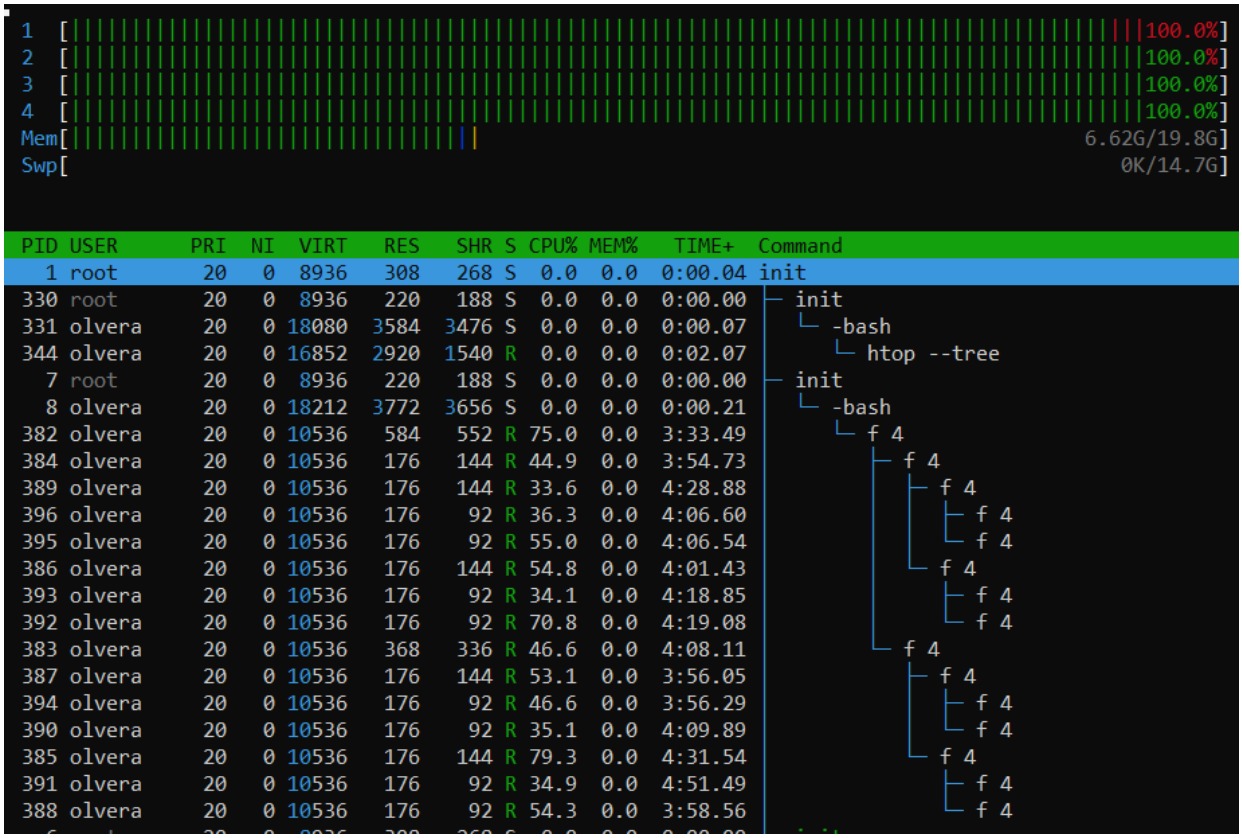
1. Crear un programa en C que permita crear un árbol binario de procesos, usando la llamada al sistema fork(). Al ejecutar el programa debe recibir como parámetro la altura de procesos que creará.

```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {

      if(argc != 2) {
        printf("No se puede pasar más de un argumento\n");
        exit(1);
      }

      int i = 0; //Es el contador para creación de los hijos
      pid_t pid; // Es el padre
      int nivel = atoi(argv[1]);

      while((nivel - 1 ) > 0 && i < 2) {
           pid = fork();

           switch(pid) {

            case -1:
                printf("Error al crear el proceso \n");
                exit(-1);

            case 0:
                printf("Soy el proceso hijo con pid: %d\n", getpid());
                nivel--;
                i =0;
                break;

            default:
                printf("Soy el proceso padre con pid: %d", getppid());
                i ++;
           }
      }
      while(1);

      return 0;
}
```

Para esta primera actividad primero declare 3 variables en donde la variable “i” se va a encargar de solo de crear dos hijos no más, la variable “pid” se va a encargar de crear los hijos y por último la variable “nivel” lo que hará es decrementar esa variable en los procesos hijos, en una unidad, y solamente llamar 2 veces a fork.

1. El primer “if” su función es que solo se le pueda pasar un solo argumento en este caso es el número 4, si le llega a pasar mas de un argumento el programa se sale inmediatamente por eso está “exit()”.
2. En el while() su función es restringir que solo se pueda crear dos hijos por nivel ya que lo estaba intentando con un for(i = 0; i < nivel; i++) pero me creaba muchos hijos entonces con while() los limitaba por eso dentro del while((nivel - 1) > 0 && i < 2) lo que hace es solo va a crear dos hijos por nivel, ya que si le llega a quitar el -1 creara muchos hijos.
3. Dentro del while el pid = fork() lo que hace es crear los hijos, después de eso se hizo un switch(pid) para que solo se puede crear los hijos, en (case -1) se utiliza por si no son los valores 0 o 1 no se pueda crear ningún proceso. En (case 0) se utiliza para crear a los hijos. Y si es default entre el padre para crear los dos hijos por nivel.

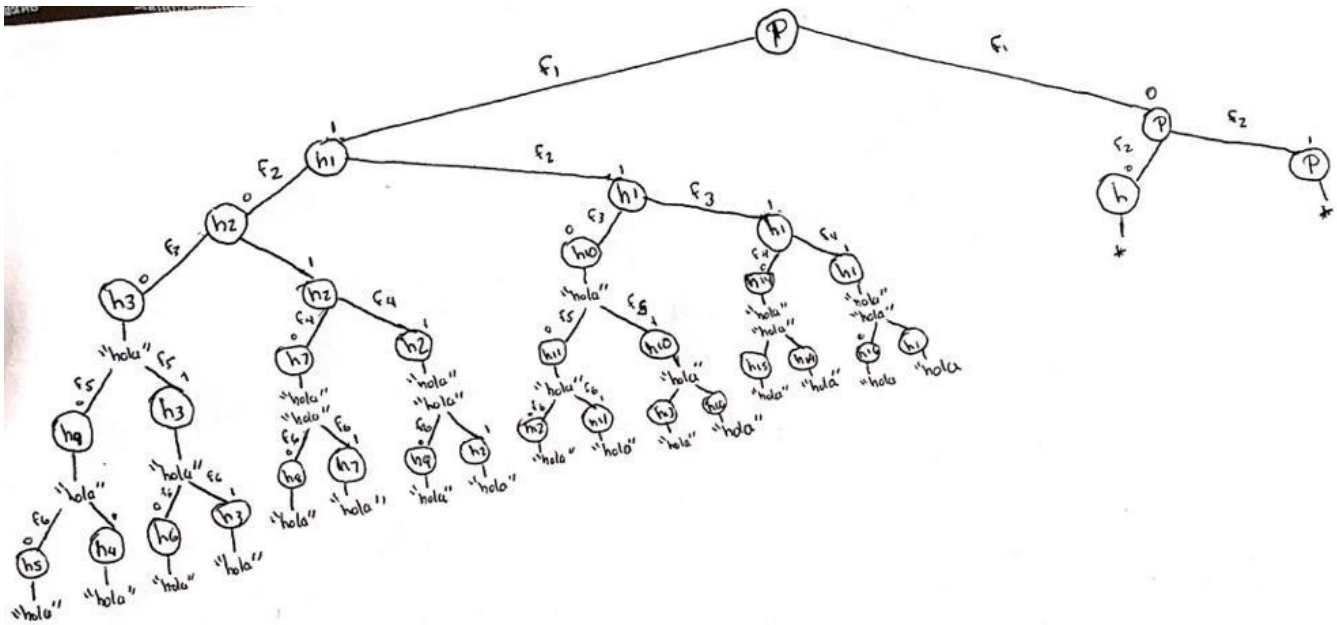
A continuación, se muestra el resultado del código:



Actividad #2

2. Indicar cuántos "hola" imprime el siguiente programa. Justificarlo con el diagrama de árbol de procesos.

```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
#include <stdlib.h>

int main(){ f1        f2        f3
     if( !fork() && fork() || fork() ){
          fork(); f4
          printf((" hola ");
     }else{
          printf(" hola ");
          fork(); f5
    }
    printf(" hola ");
    fork(); f6
    printf(" hola ");
    return 0;
}
```



*Es lo mismo que el lado izquierdo

En total son 60 "holas"

Del lado de (h1) son 30 "holas"

Y del lado de (P) son 30 "holas"

Conclusión:

Respecto a la practica la primera parte estuvo un poco complicado ya que tuve problemas para pasar un argumento ya que se me había olvidado como hacerlo en c y para que no se pudieran crear mas de dos hijos en cada nivel, pero investigando por internet me ayudo a resolver los problemas que tenía.
En la segunda actividad tuve problemas con la condición porque me estaba confundiendo al formar el árbol, pero después de analizar bien la condición pude formar el árbol.